|  |  |
| --- | --- |
|  |  |

**Task No: 7. Utilizing ‘Functions’ concepts in Python Programming.**

**7.a. You are developing a small Python script to analyze and manipulate a list of student grades for a class project. Write a Python program that satisfies the above requirements using the built-in functions print(), len(), type(), max(), min(), sorted(), reversed(), and range().**

**PROGRAM**

def analyze\_student\_grades():

# Sample data

student\_names = ["Alice", "Bob", "Charlie", "Diana"]

student\_grades = [85, 92, 78, 90]

# 1. Print a welcome message

print("Welcome to the Student Grades Analyzer!\n")

# 2. Determine and print the number of students

num\_students = len(student\_names)

print("Number of students:", num\_students)

# 3. Print the type of the student names list and the grades list

print("\nType of student\_names list:", type(student\_names))

print("Type of student\_grades list:", type(student\_grades))

# 4. Find and print the highest and lowest grade

highest\_grade = max(student\_grades)

lowest\_grade = min(student\_grades)

print("\nHighest grade:", highest\_grade)

print("Lowest grade:", lowest\_grade)

# 5. Print the list of grades sorted in ascending order

sorted\_grades = sorted(student\_grades)

print("\nSorted grades:", sorted\_grades)

# 6. Print the list of grades in reverse order

reversed\_grades = list(reversed(sorted\_grades))

print("Reversed grades:", reversed\_grades)

# 7. Generate and print a range of grade indices from 1 to the number of students

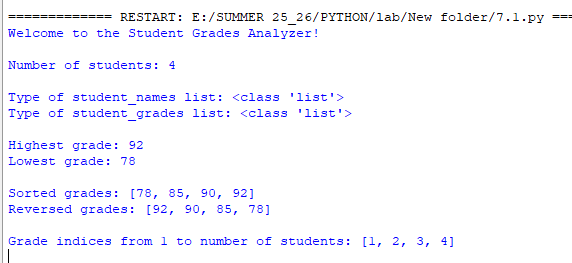
grade\_indices = list(range(1, num\_students + 1))

print("\nGrade indices from 1 to number of students:", grade\_indices)

# Run the analysis

analyze\_student\_grades()

**OUTPUT**



**7.b. You are tasked with creating a small calculator application to help users perform basic arithmetic operations and greet them with a personalized message. Your application should perform the following tasks: addition, subtraction, multiplication, division.**

**PROGRAM**

def add(a, b):

return a + b

def subtract(a, b):

return a - b

multiply = lambda a, b: a \* b

divide = lambda a, b: a / b if b != 0 else "Error: Division by zero"

# Regular function for greeting

greet = lambda name: f"Hello, {name}! Welcome to the program."

def main():

# Demonstrating the use of lambda functions

num1 = int(input("enter num1:"))

num2 = int(input("enter num2:"))

print("Arithmetic Operations:")

print("Sum of", num1, "and", num2, ":", add(num1, num2))

diff=subtract(num1, num2)

print("Difference between", num1, "and", num2, ":", diff)

if diff< 0:

print("The difference is negative!")

elif diff==0:

print("The difference is Zero!")

else:

print("The difference is Positive!")

print("Product of", num1, "and", num2, ":", multiply(num1, num2))

print("Quotient of", num1, "and", num2, ":", divide(num1, num2))

# Greeting the user

user\_name = "Alice"

print("\nGreeting:")

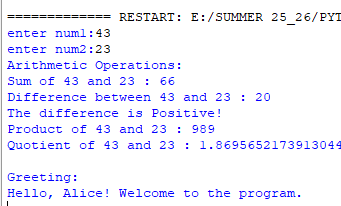
print(greet(user\_name))

# Run the main function

if \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT**



**7.C**

**Write a Python program to create a function that takes a string as input and returns its reversed form. Hint: Use a recursive function.**

**PROGRAM**

# Recursive function to reverse a string

def reverse\_string(s):

if len(s) == 0: # base case

return s

return reverse\_string(s[1:]) + s[0]

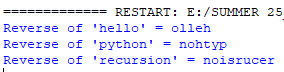
# Test cases

print("Reverse of 'hello' =", reverse\_string("hello"))

print("Reverse of 'python' =", reverse\_string("python"))

print("Reverse of 'recursion' =", reverse\_string("recursion"))

**OUTPUT**



**7.d**

**Write a Python program to analyze a list of student grades using functional programming techniques. Use lambda functions with map to modify or scale the grades, filter to select students above a certain threshold, reduce to calculate the total or average grade, and sort or sorted to arrange the grades in order. Display the results at each step, including the original grades, filtered grades, sorted grades, and the total or average.**

**PROGRAM**

from functools import reduce

# Sample data

student\_names = ["Alice", "Bob", "Charlie", "Diana"]

student\_grades = [65, 72, 58, 70]

print("Original Grades:")

print(list(zip(student\_names, student\_grades)))

# 1. Scale grades using map and lambda (e.g., add 20 internal points to each grade)

scaled\_grades = list(map(lambda x: x + 20, student\_grades))

print("\nScaled Grades (+20 points):")

print(list(zip(student\_names, scaled\_grades)))

# 2. Filter students with grades above 90 using filter and lambda

high\_achievers = list(filter(lambda x: x > 90, scaled\_grades))

print("\nStudents with grades above 90:", high\_achievers)

# 3. Calculate total and average using reduce

total = reduce(lambda a, b: a + b, scaled\_grades)

average = total / len(scaled\_grades)

print("\nTotal of scaled grades:", total)

print("Average of scaled grades:", average)

# 4. Sort grades in ascending order using sorted and lambda

sorted\_grades = sorted(scaled\_grades)

print("\nSorted Grades (Ascending):", sorted\_grades)

# 5. Sort grades in descending order using sorted and lambda

sorted\_desc = sorted(scaled\_grades, reverse=True)

print("\nSorted Grades (Descending):”, sorted\_desc )

**OUTPUT**

